Requêtes LATERALes

Vik Fearing

2013-06-13

Requêtes LATERALes

- topics
  - id integer
  - name text
- posts
  - id integer
  - topic_id integer
  - username text
  - post_date timestamptz
  - title text

Requêtes LATERALes

Afficher les cinq derniers posts par topic

Remerciements

Marc Cousin

RhodiumToad

Fonctions Window

Requêtes LATERALes

```
SELECT topics.name,
       tmp.username,
       tmp.post_date,
       tmp.title
FROM topics
JOIN
  (SELECT *
   FROM
     (SELECT *,
             row_number() OVER (PARTITION BY topic_id
                                ORDER BY post_date DESC) rownum
      FROM posts) tmpin
   WHERE rownum <= 5) tmp ON tmp.topic_id = topics.id
ORDER BY name;
```

Requêtes LATERALes

```
Sort  (cost=29180.29..29301.75 rows=48583 width=39) (actual time=346.090..346.094 rows=95 loops=1)
  Sort Key: topics.name, tmpin.rownum
  Sort Method: quicksort  Memory: 32kB
  ->  Hash Join  (cost=18663.70..24068.55 rows=48583 width=39) (actual time=244.698..345.980 rows=95 loops=1)
        Hash Cond: (tmpin.topic_id = topics.id)
        ->  Subquery Scan on tmpin  (cost=18662.25..23399.09 rows=48583 width=34)
                                    (actual time=244.653..345.905 rows=95 loops=1)
              Filter: (tmpin.rownum <= 5)
              Rows Removed by Filter: 145654
              ->  WindowAgg  (cost=18662.25..21577.23 rows=145749 width=30)
                             (actual time=244.647..335.901 rows=145749 loops=1)
                    ->  Sort  (cost=18662.25..19026.62 rows=145749 width=30)
                              (actual time=244.640..282.506 rows=145749 loops=1)
                          Sort Key: posts.topic_id, posts.post_date
                          Sort Method: external merge  Disk: 6304kB
                          ->  Seq Scan on posts  (cost=0.00..2672.49 rows=145749 width=30)
                                                 (actual time=0.009..26.076 rows=145749 loops=1)
        ->  Hash  (cost=1.20..1.20 rows=20 width=13) (actual time=0.031..0.031 rows=20 loops=1)
              Buckets: 1024  Batches: 1  Memory Usage: 1kB
              ->  Seq Scan on topics  (cost=0.00..1.20 rows=20 width=13) (actual time=0.009..0.018 rows=20 loops=1)
Total runtime: 348.059 ms
(17 rows)
```

WITH RECURSIVE

Requêtes LATERALes

```
WITH RECURSIVE
  rp AS (SELECT topic_name as topic_name, (p).*, 1 AS rcount
           FROM (SELECT t.name as topic_name,
                        (SELECT p FROM posts p
                          WHERE p.topic_id = t.id
                          ORDER BY p.post_date DESC, p.id DESC LIMIT 1) AS p
                   FROM topics t offset 0) s
                  WHERE (p).id IS NOT NULL
         UNION ALL
         SELECT topic_name, (p).*, s.rcount + 1
           FROM (SELECT rp.topic_name,
                         (SELECT p FROM posts p
                          WHERE p.topic_id = rp.topic_id
                            AND (p.post_date, p.id) < (rp.post_date, rp.id)
                          ORDER BY p.post_date DESC, p.id DESC LIMIT 1) AS p,
                        rp.rcount
                   FROM rp
                  WHERE rp.rcount < 5 offset 0) s
          WHERE (p).id IS NOT NULL)
SELECT topic_name, username, post_date, title
FROM rp
ORDER BY topic_name;
```

Requêtes LATERALes

```
Sort  (cost=927297.26..927298.99 rows=690 width=104) (actual time=571.909..571.914 rows=95 loops=1)
  Sort Key: rp.topic_name
  Sort Method: quicksort  Memory: 32kB
  CTE rp
    ->  Recursive Union  (cost=0.00..927250.93 rows=690 width=68) (actual time=12.857..571.595 rows=95 loops=1)
          ->  Subquery Scan on s  (cost=0.00..29943.71 rows=20 width=41) (actual time=12.854..117.543 rows=19 loops=1)
                Filter: ((s.p).id IS NOT NULL)
                Rows Removed by Filter: 1
                ->  Seq Scan on topics t  (cost=0.00..29943.51 rows=20 width=13) (actual time=12.848..117.501 rows=20 loops=1)
                      SubPlan 1
                        ->  Limit  (cost=1497.11..1497.12 rows=1 width=66) (actual time=5.871..5.871 rows=1 loops=20)
                              ->  Sort  (cost=1497.11..1516.29 rows=7671 width=66) (actual time=5.869..5.869 rows=1 loops=20)
                                    Sort Key: p.post_date, p.id
                                    Sort Method: top-N heapsort  Memory: 25kB
                                    ->  Bitmap Heap Scan on posts p  (cost=147.87..1458.76 rows=7671 width=66) (actual time=0.865..3.644 rows=7287 loops=20)
                                          Recheck Cond: (topic_id = t.id)
                                          ->  Bitmap Index Scan on posts_topic_id_idx  (cost=0.00..145.95 rows=7671 width=0) (actual time=0.693..0.693 rows=728…7 loops=20)
                                                Index Cond: (topic_id = t.id)
          ->  Subquery Scan on s_1  (cost=0.00..89729.34 rows=67 width=68) (actual time=4.778..90.794 rows=15 loops=5)
                Filter: ((s_1.p).id IS NOT NULL)
                ->  WorkTable Scan on rp rp_1  (cost=0.00..89728.50 rows=67 width=52) (actual time=4.775..90.764 rows=15 loops=5)
                      Filter: (rcount < 5)
                      Rows Removed by Filter: 4
                      SubPlan 2
                        ->  Limit  (cost=1339.16..1339.16 rows=1 width=66) (actual time=5.969..5.969 rows=1 loops=76)
                              ->  Sort  (cost=1339.16..1345.55 rows=2557 width=66) (actual time=5.968..5.968 rows=1 loops=76)
                                    Sort Key: p_1.post_date, p_1.id
                                    Sort Method: top-N heapsort  Memory: 25kB
                                    ->  Bitmap Heap Scan on posts p_1  (cost=66.63..1326.38 rows=2557 width=66) (actual time=0.904..3.825 rows=7668 loops=76)
                                          Recheck Cond: (topic_id = rp_1.topic_id)
                                          Filter: (ROW(post_date, id) < ROW(rp_1.post_date, rp_1.id))
                                          Rows Removed by Filter: 1
                                          ->  Bitmap Index Scan on posts_topic_id_post_date_idx  (cost=0.00..65.99 rows=2557 width=0) (actual time=0.747..0.747… rows=7670 
loops=76)
                                                Index Cond: ((topic_id = rp_1.topic_id) AND (post_date <= rp_1.post_date))
  ->  CTE Scan on rp  (cost=0.00..13.80 rows=690 width=104) (actual time=12.862..571.712 rows=95 loops=1)
Total runtime: 572.060 ms
(36 rows)
```

plpgsql

Requêtes LATERALes

```
CREATE FUNCTION n_posts (topic integer, num integer)
RETURNS SETOF posts AS
$$
DECLARE
    empty posts;
BEGIN
    RETURN QUERY
        SELECT * FROM posts WHERE topic_id = $1
        ORDER BY post_date DESC LIMIT $2;
    IF NOT FOUND THEN
        RETURN NEXT empty;
    END IF;
END;
$$
LANGUAGE plpgsql;

SELECT topics.name, (n_posts(id, 5)).* FROM topics ORDER BY topics.name;
```

Requêtes LATERALes

```
Sort  (cost=1554.87..1604.87 rows=20000 width=13)
      (actual time=19.204..19.208 rows=96 loops=1)
  Sort Key: name
  Sort Method: quicksort  Memory: 32kB
  ->  Seq Scan on topics
          (cost=0.00..126.10 rows=20000 width=13)
          (actual time=1.581..19.066 rows=96 loops=1)
Total runtime: 19.255 ms
(5 rows)
```

Tableaux

Requêtes LATERALes

```
SELECT name, (unnest(coalesce)).*
FROM
  (SELECT *,
     (SELECT coalesce(array_agg(posts),
              array[row(null,null,null,null,null)]::posts[])
      FROM
        (SELECT posts
         FROM posts
         WHERE topic_id = topics.id
         ORDER BY post_date DESC LIMIT 5) tmp)
   FROM topics
   OFFSET 0) AS tmp;
```

Requêtes LATERALes

```
Subquery Scan on tmp  (cost=0.00..88.02 rows=2000 width=41)
                      (actual time=0.218..1.454 rows=96 loops=1)
  ->  Seq Scan on topics  (cost=0.00..77.67 rows=20 width=13)
                          (actual time=0.158..1.065 rows=20 loops=1)
        SubPlan 1
          ->  Aggregate  (cost=3.81..3.82 rows=1 width=54)
                         (actual time=0.048..0.048 rows=1 loops=20)
                ->  Limit  (cost=0.42..3.75 rows=5 width=62)
                           (actual time=0.022..0.038 rows=5 loops=20)
                      ->  Index Scan using posts_topic_id_post_date_idx on posts
                                      (cost=0.42..5106.93 rows=7671 width=62)
                                      (actual time=0.020..0.030 rows=5 loops=20)
                            Index Cond: (topic_id = topics.id)
Total runtime: 1.610 ms
(8 rows)
```

LATERAL

Requêtes LATERALes

```
SELECT t.name,
       p.username,
       p.post_date,
       p.title
FROM topics t
LEFT JOIN LATERAL
     (SELECT *
      FROM posts
      WHERE topic_id = t.id
      ORDER BY post_date DESC
      LIMIT 5) p ON true
ORDER BY t.name;
```

Requêtes LATERALes

```
Sort  (cost=81.49..81.74 rows=100 width=31)
      (actual time=0.834..0.845 rows=96 loops=1)
  Sort Key: t.name
  Sort Method: quicksort  Memory: 32kB
  ->  Nested Loop Left Join  (cost=0.42..78.17 rows=100 width=31)
                             (actual time=0.064..0.589 rows=96 loops=1)
        ->  Seq Scan on topics t  (cost=0.00..1.20 rows=20 width=13)
                                  (actual time=0.011..0.017 rows=20 loops=1)
        ->  Limit  (cost=0.42..3.75 rows=5 width=30)
                   (actual time=0.017..0.025 rows=5 loops=20)
              ->  Index Scan using posts_topic_id_post_date_idx on posts
                               (cost=0.42..5106.93 rows=7671 width=30)
                               (actual time=0.016..0.023 rows=5 loops=20)
                    Index Cond: (topic_id = t.id)
Total runtime: 0.930 ms
(9 rows)
```

Autres exemples

Requêtes LATERALes

unnest() pour JSON

```
SELECT name, j->'vars'->g
FROM (VALUES ('hello', '{"vars": [1, 2, 3]}'::json),
             ('world', '{"vars": [4, 5]}'::json)) v(name, j),
     LATERAL generate_series(0, json_array_length(j->'vars')-1) g;
```

Requêtes LATERALes

Permissions des tables

```
SELECT c.oid::regclass, x.*
FROM pg_class c,
     aclexplode(c.relacl) x
WHERE c.oid = '…'::regclass;
```